Nick Seguljic

Final Report:
Stock Price Prediction

The Problem

For this project I want to see if I can predict the stock price for a given stock over a monthly basis and predict what the closing price will be each day. I want to look on a monthly basis because it is long enough to make a large return on investment, but not a large chunk of time where the confidence interval is so large the predictions become almost meaningless. The reason for looking at the low and high is to optimize profits by "buying low, and selling high" knowing full well what high and low actually are in a relative time period. For this I will employ and compare many different machine learning techniques. I will use Linear Regression, K-Nearest Neighbors, some deep learning methods like Long Short-Term Memory (LSTM) networks as it is a great method to use for large sequences of time data to make predictions about the future, ARIMA, and facebook Prophet.

The problem is complicated by the fact that it is hard to make predictions on a large time scale. It would be nice to make predictions on a year or five year basis, but the margin of error associated with this time frame is so large that it makes the model useless. Instead I will look at the time length of one month as it is a good amount of time to get a reasonable margin of errors.

Data Wrangling

After reading in the data using pandas.read_csv(), I observed the raw data head, and tail to look at the columns. The columns are the features that come from the Yahoo Finance API.

Next I analyzed the range of values of the features raw from the API:

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
1/2/19	1465.2	1553.36	1460.93	1539.13	7983100	0	0
1/3/19	1520.01	1538	1497.11	1500.28	6975600	0	0
1/4/19	1530	1594	1518.31	1575.39	9182600	0	0
1/7/19	1602.31	1634.56	1589.19	1629.51	7993200	0	0

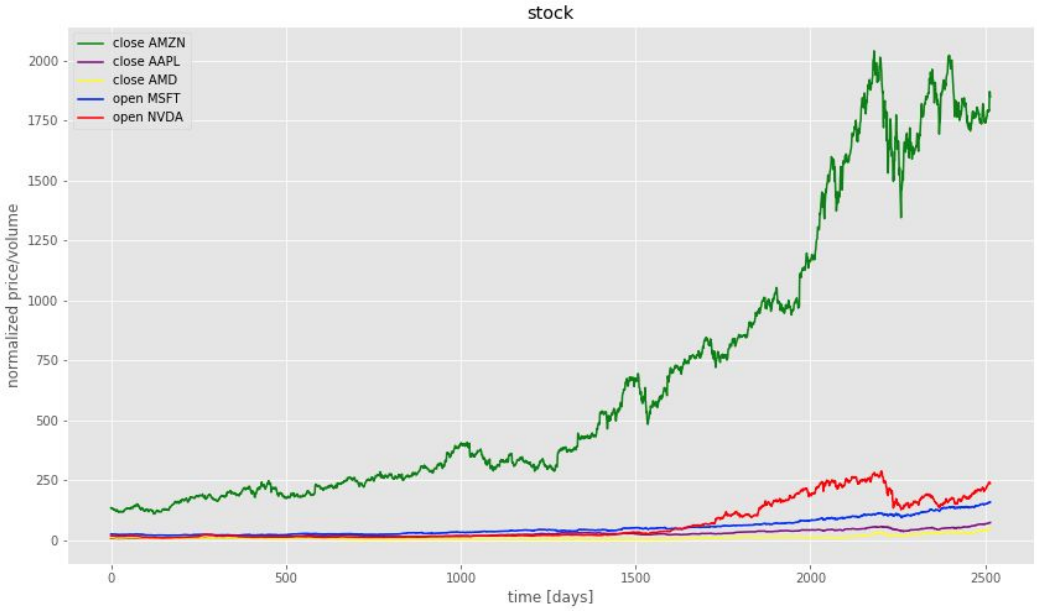
1/8/19	1664.69	1676.61	1616.61	1656.58	8881400	0	0
...	...	...	...	...	...	...	...
12/23/19	1788.26	1793	1784.51	1793	2136400	0	0
12/24/19	1793.81	1795.57	1787.58	1789.21	881300	0	0
12/26/19	1801.01	1870.46	1799.5	1868.77	6005400	0	0
12/27/19	1882.92	1901.4	1866.01	1869.8	6186600	0	0
12/30/19	1874	1884	1840.62	1846.89	3674700	0	0

Next I added one feature for each stock: return. Return is percent change from one closing price to the next for a given stock.

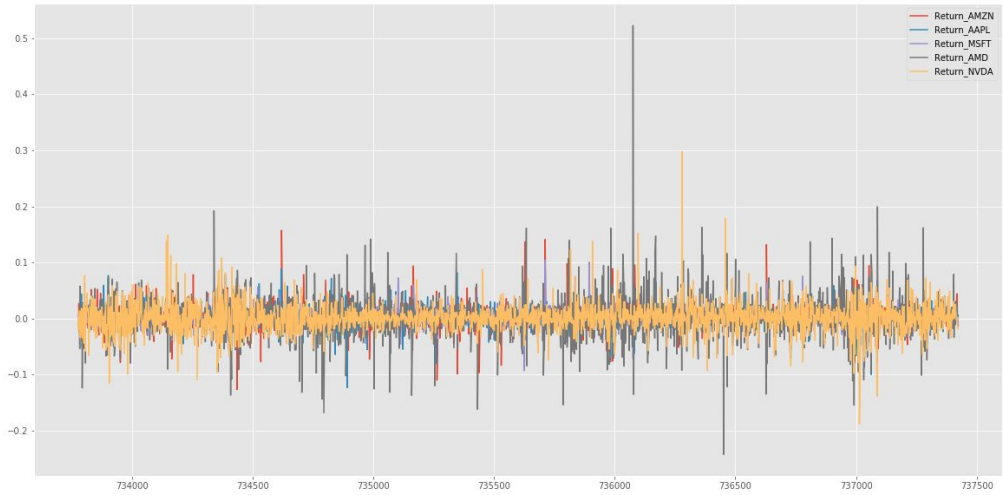
Exploratory Data Analysis

The stock data provided by Yahoo from their API is readily available and goes back to 1970. I want to know if I can predict the stock price for a given stock over a monthly basis and predict what the closing price will be each day. I will look specifically at the stocks AMZN, AAPL, MSFT, NVDA, and AMD.

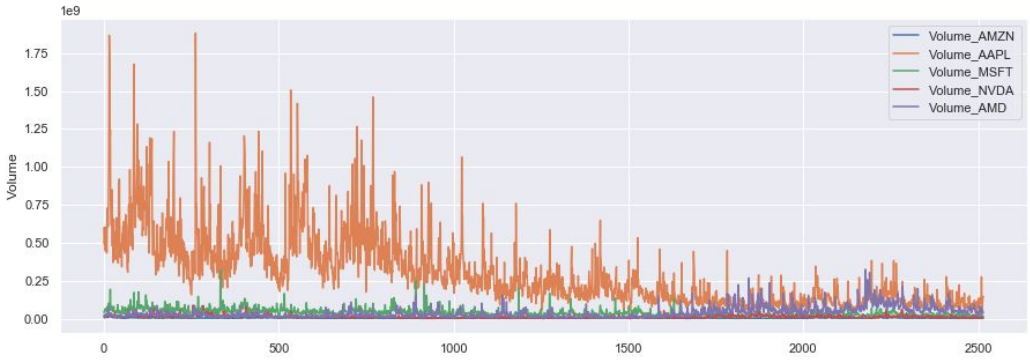
First I wanted to look at how they all compare to each other over the ten year period:



Let's take a look how the returns vary over time for each stock:

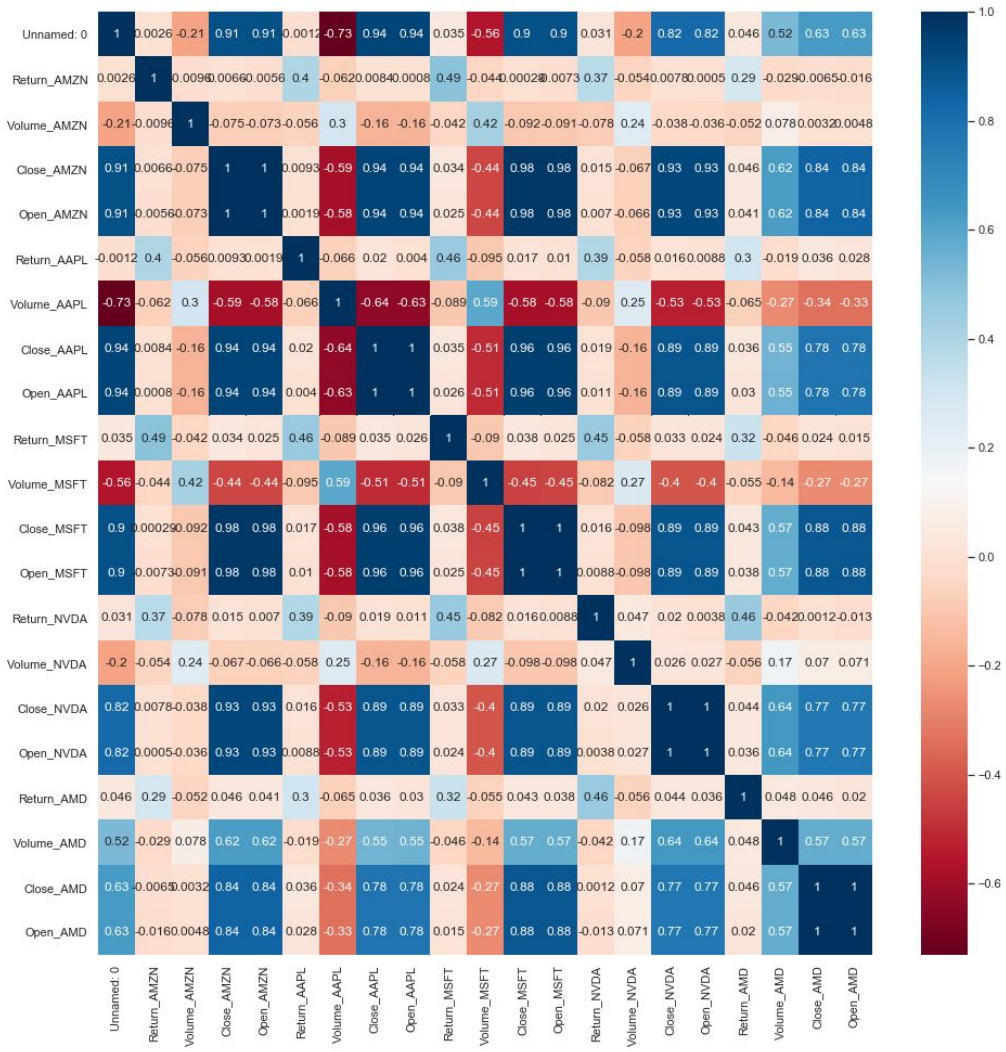


Now let’s look at the volume of each stock:

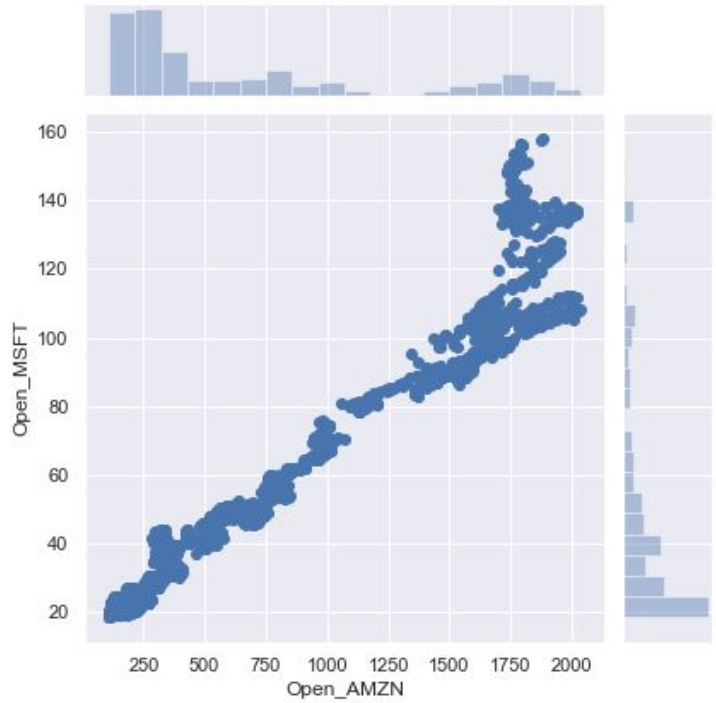


AMZN is a winner in terms of price and growth, but it has seen quite a dip in volume trading in the past five years. In terms of returns it looks like the stocks all perform around the same.

Let’s take a look at how the features vary with each other using a heatmap:

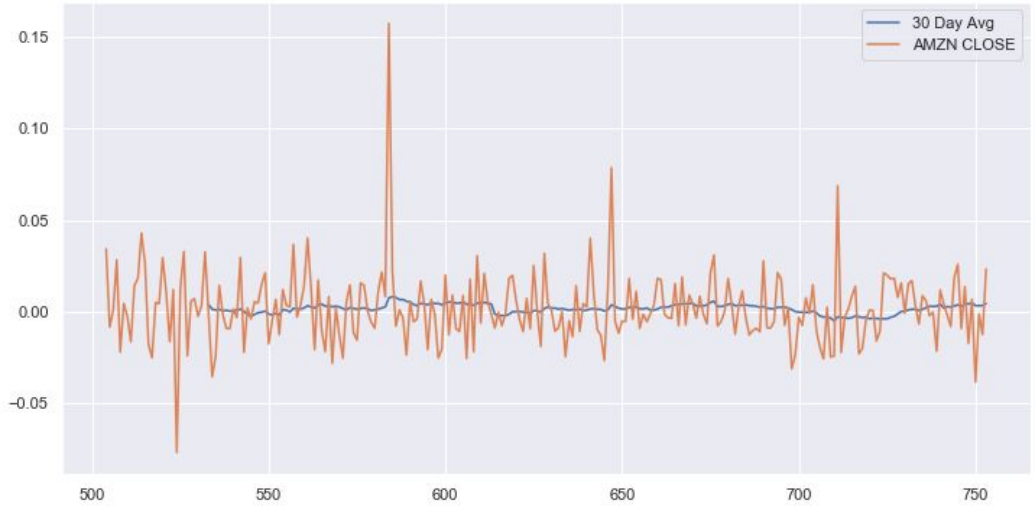


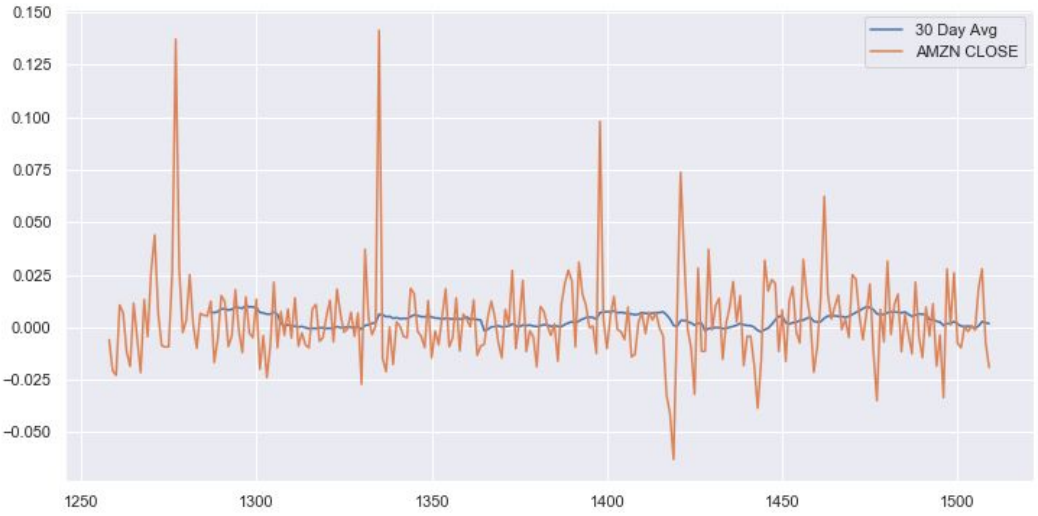
Most of the correlation are obvious, the features of MSFT correlate highly with features of MSFT, and this is true of all the stocks. One interesting item to note is that Open_MSFT and Open_AMZN which has a coefficient of 0.98. Below is a visualization of how they vary:

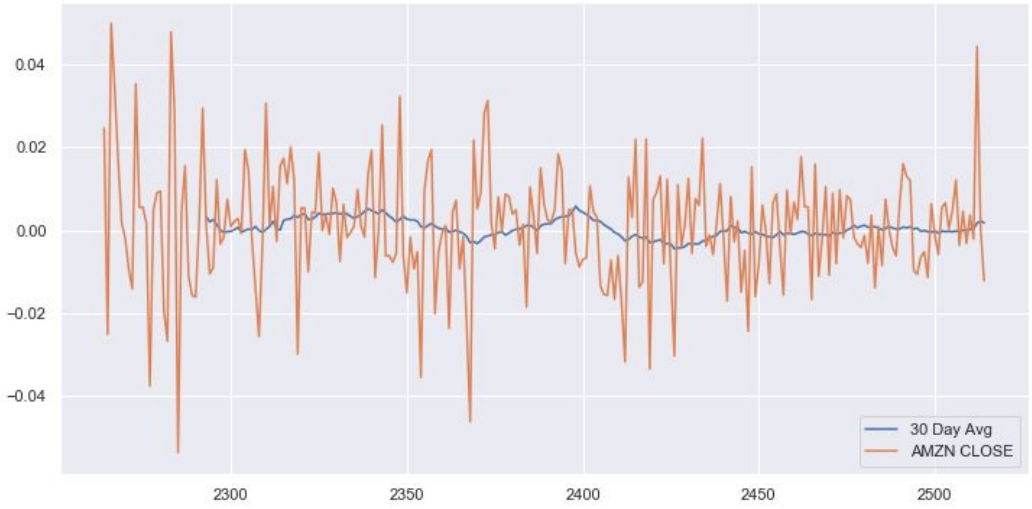


It makes sense that they correlate well. As mentioned before the returns of all the stocks seem to be similar to each other, MST and AMZN probably just happen to grow at about the same rate, leading to this linear relationship. It likely does not exist with something like NVDA because NVDA is slightly more volatile, and these two are slightly more stable. Knowing that stocks grow similar to each other means that the same algorithms will likely perform similarly for the given stocks. As speculated before, it would likely make sense to bucket stocks based on their performance(low,medium, and high for instance) and I think that different models will be better at predicting these different buckets. Knowing this about MSFT and AMZN means I can look to prove this theory if they perform similarly with each type of algorithm.

The last item I wanted to look at was how and if seasonality affects AMZN. Here are visuals for 3 different years, 2012, 2015, and 2019. The graphs are AMZN returns, and another line that shows the average returns over the thirty day period at the same time. This is done in order to see if certain months always have an above average return.


30 Day Avg
AMZN CLOSE
0.15
0.10
0.05
0.00
−0.05
500
550
600
650
700
750


30 Day Avg
AMZN CLOSE
0.150
0.125
0.100
0.075
0.050
0.025
0.000
−0.025
−0.050
1250
1300
1350
1400
1450
1500



It does not look like any seasonality exists for AMZN.

Model Selection

In this section I will look at the predictive power of machine models and neural networks to try to understand what models will be best for predicting stock prices on a monthly basis. I will look to see what models perform the best and look to see how different stocks perform with different models. But first I will give an introduction to the methods I will use. I will use the standard models such as Linear Regression and kNN, but I will also use time-series based deep learning algorithms.

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. Unlike standard feed forward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). — Wikipedia

In essence an LSTM is a powerful time-series model that can predict any number of steps into the future. It can be thought of as a series of cells connected with each other through feedback loops that allow for the prediction of single data points, or sequences of data. LSTM cells have 5 components that allow it to make short and long term predictions based on previous data. The cells contain the cell state(the internal memory where short and long term memories are), the hidden state(output state information calculated from the current input, it can decide to only retrieve the short or long-term or both types of memory stored), the input gate(the gate that decides what and how much information is made available), the forget gate(the gate that decides how much information from the input flows to the cell), and the output gate(the gate that decides how much information from the current cell state flows into the hidden state). All of the components working together are what allow models to make long and short term predictions.

Why use it on stock data?
They're able to store past information, and the current price of a stock is dependent on the previous price of a stock. Not only this but, past prices from years out make a difference(long term), and recent prices(short term) also make a difference. Using both is key to predict stock prices.

ARIMA (AutoRegressive Integrated Moving Average) model is a generalization of an autoregressive moving average (ARMA) model. Both of these models are fitted to time series data either to better understand the data or to predict future points in the series (forecasting). ARIMA models are applied in some cases where data show evidence of non-stationarity, where an initial differencing step (corresponding to the "integrated" part of the model) can be applied one or more times to eliminate the non-stationarity. — Wikipedia

The model takes in three inputs: p, the number of autoregressive lags, d, order of differencing, and q, number of moving average lags.

ARIMA is much better used on short term time predictions.

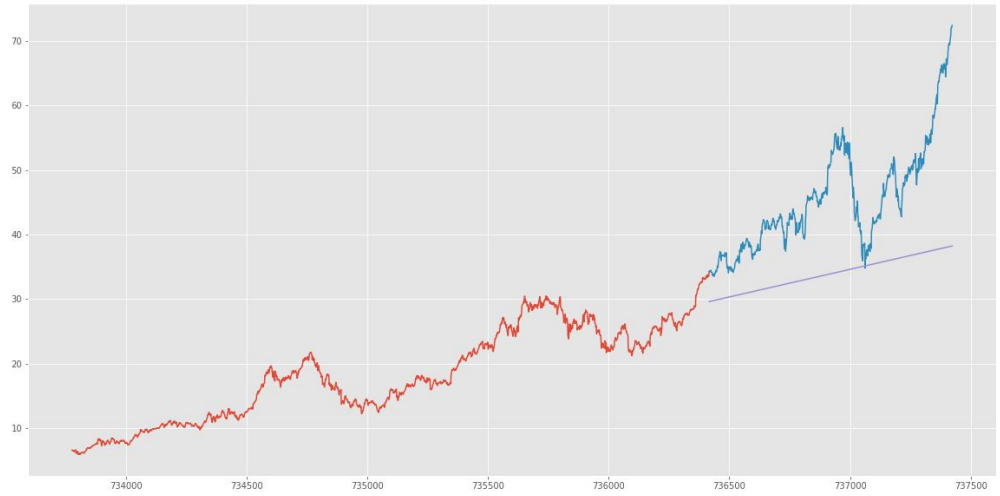
The first model I looked at was Moving Average. Here are visuals for AMZN and AAPL.

Moving Average

In terms of figuring out the general trend of the stock data, the moving average method did okay, but it failed to see the full extent of the increase in the price, and that is not good. We definitely wouldn't want to use this method for actual algorithmic trading.

Linear Regression

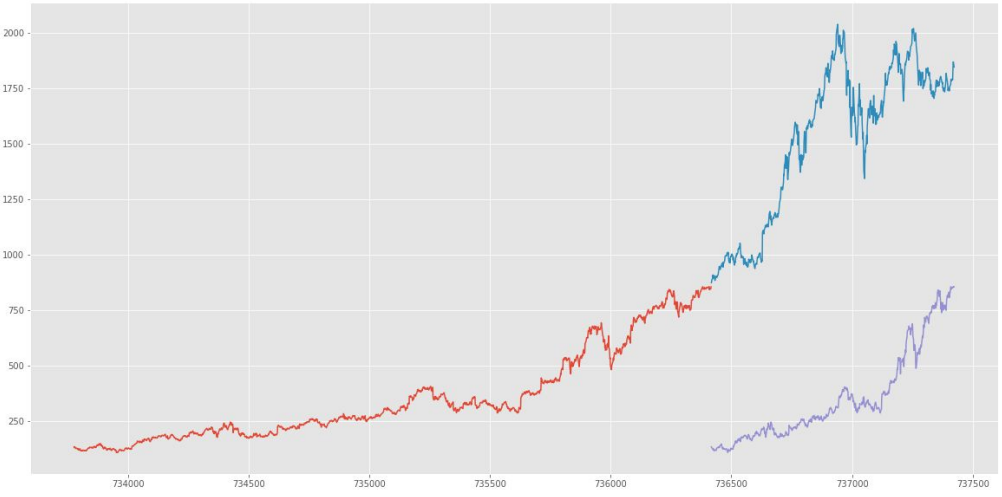
Next up I looked at linear regression for the same two:



Linear regression performed much worse with a much higher RMSE, and a line of a smaller slope.

This may not be all bad though, linear regression is likely better looking at the overall trend from the past 10 years and predicting, whereas the Moving average method will be looking at more short term data. In this instance Moving Average did better because AMZNs price shot up, but had that not happened it may have done worse. This could mean with more stable stock or time series data linear regression could perform better.

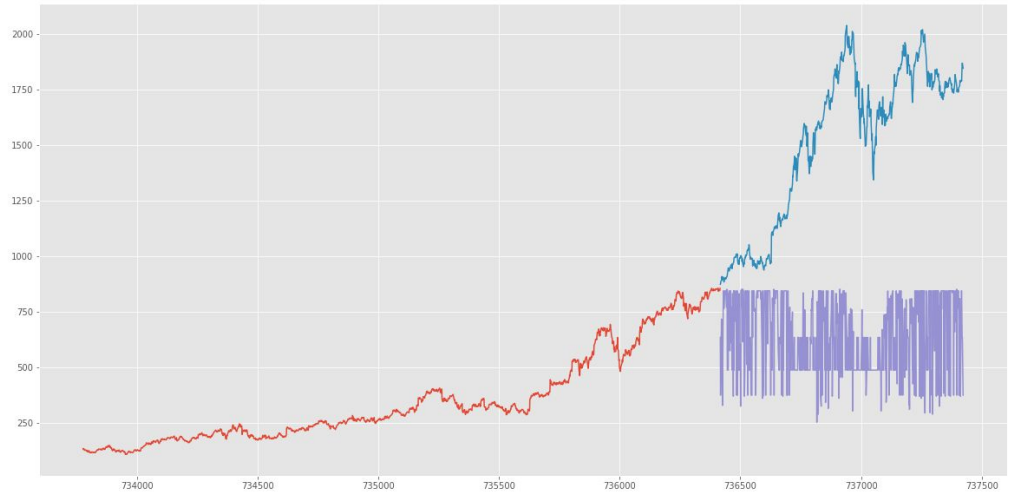
KNN



KNN is way off and has a high RMSE. It seems to be better at predicting the peaks and troughs of the data, but the error is much too high. It also seems to be able to predict the data is going up, but the data is overfitted.

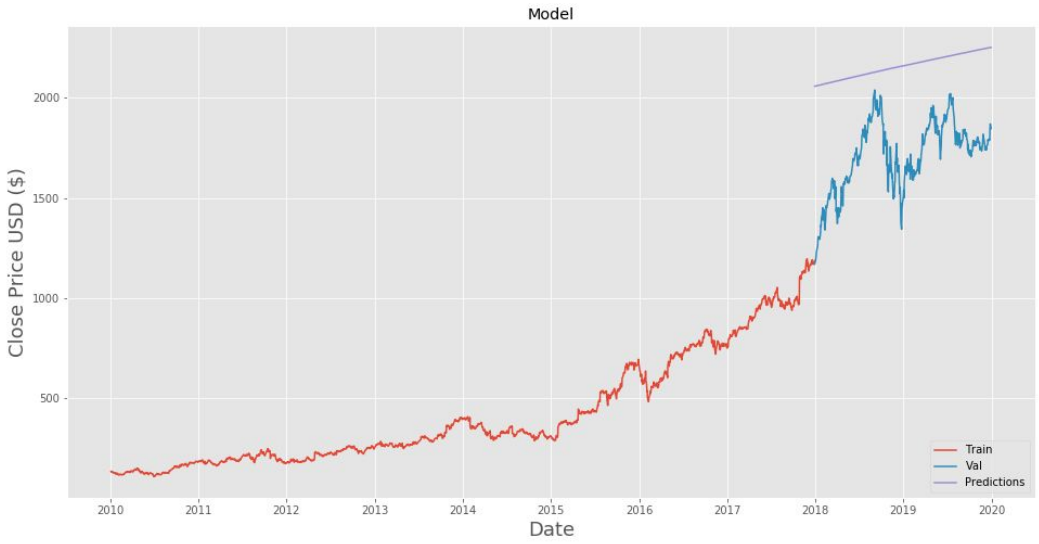
Not looking to good, let's try XGBoost

XGBoost

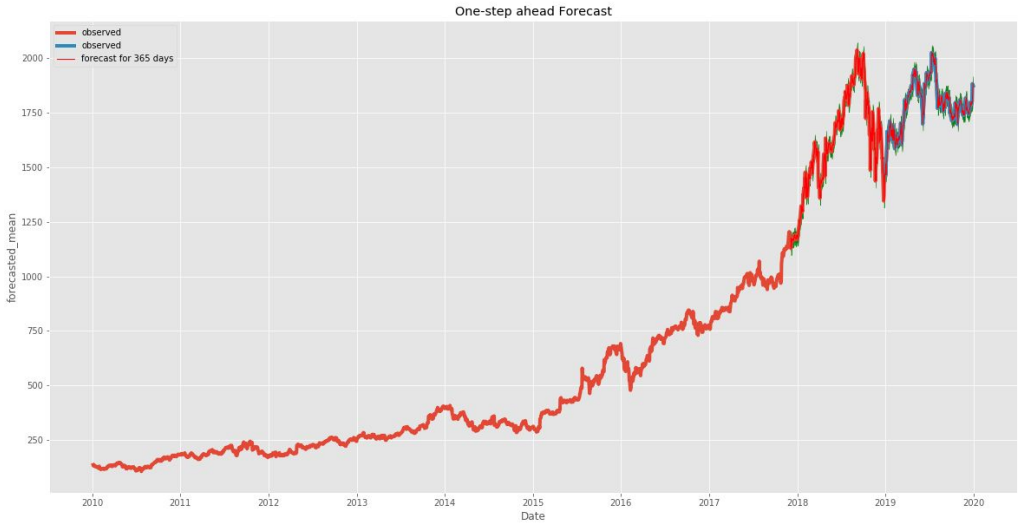


XGBoost looks like it has lower error, but is much too volatile and has no apparent trend. Let's see what it looks like for another stock.

LSTM



This is by far the worst model, it has the worst RMSE, and the predicted data is just a flat line with no trend at all. Let's see if ARIMA can do better.



ARIMA is by far the best model when looking at error metrics and it's graph, and kNN is the worst. Now the question is why is this so? ARIMA is better at short term forecasting, where LSTM has an edge with the long term, so it is possible that if the length of time I wanted to predict were bigger that we would see LSTM start to outperform. Below is the chart for RMSE of all the models, as mentioned ARIMA does well, but is still only second to linear regression. While linear regression has the lowest RMSE and predicts trends well, it still does not fit the data well according to the graph.

AMZN Model	AMZN RMSE
Moving Average	682.965744
Linear Regression	13.296866
KNeighborsClassifier	1193.898429
XGBoost	16.453802
LSTM	352.344766
ARIMA	25.460426

Major Findings

ARIMA had the second lowest RMSE, and generally predicted the trend well for the stock prices. It is important to note that it was still nowhere close to the actual price at any point. This leads me to conclude and suggest from this work that only trends can be gleaned from this work. It seems that predicting stock price is too difficult but using this to predict trends and then designing a portfolio around what has the most potential growth would be best for investors.

I also found that the more growth has, the more off the prediction will be. It is not done here, but if one took a stock from something like Coca-Cola, it would most likely be able to predict trends quite well since it is stable, but a stock like NVDA is all over the map sometimes, usually in the upward direction. This adds a layer of difficulty because not only can one not accurately predict the stock price, it becomes difficult to predict an accurate trend as well.

Further Research

For further research I would look at using different models to lower error, such as Facebook Prophet or other neural network based models.
I would also recommend using these models as a way to get the best possible information on trends and then using something like a Monte Carlo simulator to best predict how one should invest their money based on potential growth and risk.